U.S. National Library of Medicine
National Center for Biotechnology Information

Five Examples for NCBI APIs

Teaching NCBI Resources

Through Use Cases And Examples

Introduction

Large volumes of biomedical research literature and datasets are available from the public databases maintained by NCBI. In addition to the access through the web interface, NCBI also provides Application Programming Interfaces (APIs) to help with downloading and processing of these data sets. A comprehensive document describing the Entrez Programming Utilities (EUtils), is at:
https://www.ncbi.nlm.nih.gov/books/NBK25501/

The Entrez Direct (EDirect) package is a set of scripts that access EUtils service. These allow you to build custom workflows without having to write your own programs. The powerful XML parser, xtract, allows you to easily extract desired values from the retrieved records. A detailed description of this package is at:
https://www.ncbi.nlm.nih.gov/books/NBK179288/

In this booklet, you will find examples of sample queries using EDirect and the PubChem Power User Gateway API (PUG REST) in a Unix environment. All examples explore aspects of human creatine kinase genes, such as the various symbols, finding citations, related proteins, and variations. To gather information about this gene, you will begin to learn how to use EDirect, the E-utilities on the Unix/Linux command line. This is a great way to start using command line programs and allows you to begin exploring computational biology and bioinformatics. This document assumes that you have EDirect installed and know how to get to the Unix environment. You can also refer to the following pages for general overview and instructions on installation
https://dataguide.nlm.nih.gov/edirect/overview.html
https://dataguide.nlm.nih.gov/edirect/install.html

Linux command lines and URL calls used in this handout are available as a text file at:
https://ftp.ncbi.nlm.nih.gov/pub/factsheets/booklet_edirect.txt

Use the commands from this file instead of copying those given in the textboxes of this document - PDF generation process may introduce hidden characters or convert certain characters to non-ascii counterparts.

Table of Content

Exercise 1: Find the official gene symbols of human creatine kinase genes, and any other names that have been used for them.

Goal

From the name of the gene, use EDirect to find the official gene symbol, other gene symbols used over the years, and full gene or protein name. Specifically finding current protein-coding genes on the human chromosomes.

Solution

The series of commands (color-coded to separate individual steps):

1) searches Gene for the string ‘creatine kinase[Gene/Protein Name] AND human[Organism] AND "genetype protein coding"[Properties] AND alive[prop]‘
2) retrieves the document summary (docsum) for each of the search results
3) extracts official gene symbol, other aliases, and protein names, and
4) writes the extracted information to a text file (aliases-gene.txt).

```
$ esearch -db gene -query '"creatine kinase"[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties]' | \
efetch -format docsum | \
xtract -pattern DocumentSummary -element Name OtherAliases OtherDesignations > aliases-gene.txt
```

Sample output

The table to the right contains a sample of the contents within the output file (aliases-gene.txt), broken down by column: the official gene symbol from the document summary is in the first column, the list of other aliases (separated by commas) are in the second column, and the list of protein names (separated by pipe "|" characters) are in the third column.

Column	Value
Official Symbol	CKM
Other Aliases	CKMM, CPK-M, M-CK
Other Designations (pipe-separated)	creatine kinase M-type|creatine kinase M chain|creatine kinase, muscle|creatine phosphokinase M-type

Explanation

```
$ esearch -db gene -query '"creatine kinase"[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties])' | \
```

The first command uses esearch to search Gene (***-db gene***) for our field-limited search query (***"creatine kinase"[Gene/Protein Name]) AND “genetype protein coding”[Properties] AND human[Organism] AND alive[prop]"***). The "|" character pipes the results from esearch into the next command, and the "\" character at the end of line allows the commands to continue on the next line, for easier to-read formatting.

```
  efetch -format docsum | \
```

The second line takes the esearch results from the first command and uses efetch to retrieve the Document Summary (***-format docsum***) for each of the records, an XML wrapped summary of the important fields of a gene record. Gene database also supports other retrieving formats, such as full_report, gene_table, and xml.

```
xtract -pattern DocumentSummary -element Name OtherAliases OtherDesignations > aliases-gene.txt
```

The third command uses the xtract to retrieve only values from selected fields with the relevant information we need and output them in a tabular format. The -pattern option indicates that we should start a new row for every gene record. The command then extracts each gene’s official symbol, other aliases, and other designations (***-element Name OtherAliases OtherDesignations***). The output contains a list of genes, one record per line.

The last generic unix shell command writes (redirects) the output (**>**) to a file (***aliases-gene.txt***).

Exercise 1 (cont.)

Where do all the gene symbols come from? The NCBI Gene database receives the official gene symbol designations from the organism-specific gene nomenclature committee. The Human Genome Organization (HUGO) has a Gene Nomenclature Committee (HGNC) that sets the standards for human gene nomenclature. While the aliases are added to Gene records by RefSeq curation staff based on literature reviews.

Exercise 2: Get a list of all the PubMed citations for the human creatine kinase genes.

Goal

Find literature for a gene, regardless of gene symbol or alias input. We will use EDirect to connect the gene list from Case 1 to PubMed ids (PMIDs).

Steps

The series of commands:

1) searches for gene records specified in Exercise 1
2) links from these gene record to PubMed
3) retrieves the DocumentSummary XML for each of the PubMed citation
4) extracts the PMID, journal title abbreviation and article title, and
5) writes the data to a text file (gene2pubmed.txt)

```
$ esearch -db gene -query '"creatine kinase"[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties]' | \
  elink -db gene -target pubmed | \
  efetch -db pubmed -format docsum | \
  xtract -pattern DocumentSummary -element Id Source Title > gene2pubmed.txt
```

Sample Output

The output text file (***gene2pubmed.txt***) contains three columns: PMIDs, source journal of the article, and the article title. The table to the right shows an example entry from the file, broken down by column.

Column	Value
PMID	28514442
Source	Nature
Title	Architecture of the human interactome defines protein communities and disease networks

Explanation

```
$ esearch -db gene -query '"creatine kinase"[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties]' | \
```

The first command use the same esearch line from the first case, where we search in Gene database (***-db gene***) for our genes (***-query '"creatine kinase" [Gene/Protein Name] AND human[Organism] AND "genetype protein coding"[Properties] AND alive[Properties]'***).

```
  elink -db gene -target pubmed | \
```

The second command links (***elink***) gene records (***-db gene***) from the first command to their relevant pubmed records (***-target pubmed***).

```
  efetch -db pubmed -format docsum | \
```

The third command retrieves these citation records from pubmed (***-db pubmed***) in DocumentSummary format (***-format docsum***).

```
  xtract -pattern DocumentSummary -element Id Source Title > gene2pubmed.txt
```

The fourth command parses docsum XML to extract PMID, source journal, and title (***-element Id, Source, Title***) in the docsum XML. The last command writes the output to a file (***> gene2pubmed.txt***).

Exercise 3: Locate the creatine kinases genes on the human chromosomes and retrieve their genomic sequences.

Goal

Gather chromosomal information for each of the gene records found in Case 1. You can then use this information to download the genomic sequences.

Steps

The set of commands:

1) searches for gene records specified in Exercise 1
2) retrieves the document summary (docsum) for each of the search retrieved records
3) extracts gene symbol, gene id, chromosome accession, chromosome, chromosomal start/stop coordinates, and
4) writes the extracted information to a text file (chrpos-gene.txt)

```
$ esearch -db gene -query '"creatine kinase[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties]' | \
  efetch -format docsum | \
  xtract -pattern DocumentSummary -element Id, Name \
-block GenomicInfoType -element ChrAccVer, ChrLoc, ChrStart, ChrStop > chrpos-gene.txt
```

Sample Output

The output file (chrpos-gene.txt) contains the Gene ID, Name, Chromosome Accession, Chromosome, Chromosoaml Start, Chromsomal Stop in a tab-delimited format. The table to the right contains the first few lines from the result file.

GeneID	Official Symbol	Chromosomal Accession	Chro-mosome	Start Coordinate	Stop Coordinate
1158	CKM	NC_000019.10	19	45322976	45306412
1152	CKB	NC_000014.9	14	103522858	103519657
1159	CKMT1B	NC_000015.10	15	43592856	43599405

Details

```
$ esearch -db gene -query '"creatine kinase"[Gene/Protein Name] AND human[Organism] AND "genetype
protein coding"[Properties] AND alive[Properties]' | \
```

The first command is the same as in Exercise 1 and 2. Similar to Exercise 1, the resulting gene IDs (UIDs) are used to fetch the corresponding gene record.

```
  efetch -db gene -format docsum | \
```

The second command uses the efetch program to retrieve from the Gene database (***-db gene***) all the records found by the esearch command in docsum format (***-format docsum***). This docsum format also includes a “GenomicInfoType” section which contains the information we need.

```
  xtract -pattern DocumentSummary -element Id  Name \
-block GenomicInfoType -element ChrAccVer ChrLoc ChrStart ChrStop \ > chrpos-gene.txt
```

The third command invokes the xtract program to extract only the elements we need, and display those elements in a tabular format. The ***-pattern DocumentSummary*** indicates that we should start a new row for each gene record present in the docsum result. The ***-element Id Name*** argument extracts the GeneId and official symbol. The ***-block GenomicInfoType*** argument looks specifically at the GenomicInfoType section of each record, and then use the ***-element ChrAccVer ChrLoc ChrStart ChrStop*** to extract the chromosome accession, chromosome, and start/stop coordinates.

The last command ***> chrpos-gene.txt*** writes the result to the named file, which is a tab-delimited file with gene ID followed by the gene symbol, chromosome accession, chromosome and start/stop coordinates in separate columns.

Exercise 3 (cont.)

Knowing the accessions and positions, you can retrieve the genomic regions from Nucleotide database. Since Gene uses a zero-based coordinate with first nucleotide numbered as zero, while Nucleotide database coordinates are 1-based, we need to shift the coordinates accordingly by incrementing the gene coordinates. Efetch function takes care this for us if we provide the coordinates through -chr_start and -chr_stop arguments.

```
$ efetch -db nucleotide -id NC_000015.10 -format fasta -chr_start 43592856 -chr_stop 43599405 \
> chmt1b.fa
```

The efetch command above retrieves the sequence for CKMT1B gene (***-id NC_000015.10 -chr_start 43692643 -chr_stop 43699221***) in FASTA format (***-format fasta***) from the nucleotide database (***-db nuccore***).

Further Exploration

Once you become more comfortable with the Unix command line, you can start using standard Unix commands, such as ***xargs***, as shown below, to retrieve all the regions using a single command line.

```
$ cat chrpos-gene.txt | xargs -n 6 sh -c 'efetch -db nucleotide -format fasta -id $2 -chr_start $4
-chr_stop $5' > gene_seq.fa
```

Exercise 4: Gather variants in the CKMT1A gene (id:548596) and their corresponding clinical significance.

Goal

While reading the research literature, you notice discussions about variants within the creatine kinase genes that have clinical implication. Now, you want a list of all the known variants mapped to one of the gene along with their clinical significance. Specifically, this should be a table of variant ids within a gene, clinical significance, date of when last evaluated, and review status of clinical significance.

Steps

Since we already know the Gene ID for CKM (548596) from previous exercises, we can start with the following steps:

1) searches for the gene record using the field-limited gene symbol in the query
2) followed by linking the gene id to ClinVar records
3) fetch the linked ClinVar ids in XML format, and
4) extracts the ids, description of clinical significance, when last evaluated, and status of review
5) writes the output to a file (gene2clinvar.txt)

```
$ esearch -db gene -query "CKMT1A[gene]" | \
  elink -target clinvar | \
  esummary | \
  xtract -pattern DocumentSummary -ID Id -block clinical_significance -element
"&ID",description,last_evaluated,review_status > gene2clinvar.txt &
```

Sample Output

The query returns a table of variants in the CKMT1A gene (geneid:548596) along with their corresponding clinical significance.

Variation ID	Clinical Significance	Date of Last Review	Review Status
602119	Benign	2012/07/03 00:00	no assertion criteria provided
614520	Pathogenic	2017/01/05 00:00	no assertion criteria provided
155042	Uncertain significance	2012/10/23 00:00	no assertion criteria provided
148856	Likely pathogenic	2011/11/04 00:00	no assertion criteria provided

Exercise 4 (cont.)

Explanation

The first command searches (***esearch***) for a specific gene record (***-db gene –query “CKMT1A[gene]”***).

```
$ esearch -db gene -query "CKMT1A[gene]" | \
```

The second command takes the search retrieved geneid and uses that as input to link (***elink***) from gene (implied from the first command) to the target ClinVar database (***-target clinvar***) to find connected ClinVar records.

```
  elink -target clinvar | \
```

The third command gets the document summary of these ClinVar records (***esummary***, with the database implied from the previous step and record format in the default XML).

```
  esummary | \
```

The fourth command uses the XML parser (***xtract***) to retrieve only the elements we need from the XML output, and display those elements in a tabular format.

```
  xtract -pattern DocumentSummary -ID Id -block clinical_significance -element
"&ID",description,last_evaluated,review_status
```

1) The ***-pattern DocumentSummary*** argument indicates that we will start a new row for each ClinVar record. The ***-ID Id*** extracts the ClinVar Variation ID and assigns its value to a shell variable, which is indicated by a hyphen followed by a string of capital letters (***-ID***) or digits. You can reference the variable value by prefixing an ampersand to the variable name (***&ID***) to display the value multiple times as needed.

2) The –***block clinical_significance*** is the block name, from which we need to extrac the clinical significance information for each of the ClinVar variantion record

3) The ***-element "&ID",description, last_evaluated,review_status*** portion of the command gets Variation ID (referencing the value of a shell variable), then parses out variant clinical significance from the description field, last evaluation date from the last_evaluated field, and the review status from the review_status field. The “,” separator separates the values by tab.

The last command redirects (**>**) the output to a file named *gene2clinvar.txt*.

```
  > gene2clinvar.txt &
```

Exercise 5: Compare the properties of creatine and phosphocreatine.

Goal

The PubChem family of databases (Compound, Substance and BioAssay) provide information on chemical compound and their biological activities. NCBI PubChem database offers a fetch service through its own Power User Gateway (PUG) setup (https://pubchem.ncbi.nlm.nih.gov/pug/pughelp.html). The PUG REST service is a set of structured URL-based calls (https://pubchem.ncbi.nlm.nih.gov/rest/pug/) for accessing and retrieving information from the PubChem set of databases.

In this exercise, we use PUG REST to gather information about the PubChem ID (CID), IUPAC name, molecular formula, molecular weight, and structure for two chemical compounds, creatine and phosphocreatine.

Steps

The PUG REST calls

1) search for CID(s) within the compound database from the name given
2) fetch the PubChem Compound record(s) from the found CID(s)
3) extracts the chemical properties and writes the data to a CSV file

Exercise 5 (cont.)

For creatine, the URL request is:
https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/name/creatine/property/
IUPACName,MolecularFormula,MolecularWeight,CanonicalSMILES/CSV

For phosphoecreatine, the URL request is:
https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/name/phosphocreatine/property/
IUPACName,MolecularFormula,MolecularWeight,CanonicalSMILES/CSV

Sample Output

An output for each of the input commands will be a CSV file that contains a table of the asked information for the chemical compound.

Explanation

https://pubchem.ncbi.nlm.nih.gov/rest/pug/
The first part of the URL is the base URL for PubChem PUG REST service.

The second subset specifies the target database, the input format, followed by the input query (the name of the compound in this case):
/compound/name/creatine/

CID	IUPACName	MolecularFormula	MolecularWeight	CanonicalSMILES
586	2-[carbamimidoyl(methyl)amino]acetic acid	C4H9N3O2	131.135	CN(CC(=O)O)C(=N)N
CID	IUPACName	MolecularFormula	MolecularWeight	CanonicalSMILES
9548602	2-[methyl-[(E)-N'-phosphonocarbamimidoyl]amino]acetic acid	C4H10N3O5P	211.114	CN(CC(=O)O)C(=NP(=O)(O)O)N

The third subset specifies the type of information requested, and the specific fields of interest in the form of a comma-separated list, followed by the output format desired:
/property/IUPACName,MolecularFormula,MolecularWeight,CanonicalSMILES/CSV

In addition to CSV, you can also request TXT, XML, or PNG (for the 2D structure image) format:
https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/name/phosphocreatine/PNG

Other References

NCBI Learn page provides access to webinars, workshops, and tutorials:
https://www.ncbi.nlm.nih.gov/home/learn/

NCBI webinars and other video tutorials are available from its Youtube channel:
https://www.youtube.com/user/NCBINLM

EDirect help manual is available from Bookshelf:
https://www.ncbi.nlm.nih.gov/books/NBK179288/

A collection of short factsheets and how-to's are listed here:
http://bit.ly/ncbi_factsheets

PubChem help is online at:
https://pubchemdocs.ncbi.nlm.nih.gov/

This blogpost provide information on the recently implement API key:
https://ncbiinsights.ncbi.nlm.nih.gov/2017/11/02/new-api-keys-for-the-e-utilities/

Please send your questions and comments to:
info@ncbi.nlm.nih.gov